SUL - Simple User Interface Language 1.0

1 May 2007

This version:

http://www.openxup.org/TR/sul-20070501.pdf

Latest version:

http://www.openxup.org/TR/sul.pdf

Editor(s):

Jin Yu, OpenXUP.org, jyu@openxup.org.

Abstract

The Simple User Interface Language (SUL) is an XML-based language for modeling rich user interfaces. It contains a set of UI components, events, and resources.

Feedback and comments are welcome and may be sent to info@openxup.org.

Table of Contents

1. Introduction

The Simple User Interface Language (SUL) is an XML-based language for modeling rich user interfaces. Its goal is to make the rich UI components found in desktop GUI toolkits (e.g. Windows Forms and Java Swing) available to web applications.

SUL models UI **components** such as buttons and labels as XML **elements**, and UI **properties** such as color and font as XML **attributes**.

For UI **events**, SUL uses a delegation-based [1] event model; therefore, events do not propagate up or down in the document hierarchy. All events have a type, and some events may have event details, enclosed by the <xup:detail> element. For example, the event detail of a mouse click event specifies the coordinates of the mouse pointer, which button was clicked, and modifier keys. Furthermore, SUL also supports fine-grained event selection; that is, in addition to selecting event by its type, applications may specify event masks (via the <xup:mask> element) to further refine event selection criteria. For instance, for a mouse click event, an application may specify that the event should be fired only if the "control" modifier key was held down.

Finally, for UI **resources** such images and files, SUL models them as XML elements.

Note that SUL elements do not contain mixed XML content. That is, an SUL element's content is either text, a list of children elements, or empty.

1.1 Terminology

The key words **may**, **must**, and **should** in this document are to be interpreted as described in RFC 2119 [2].

An implementation is not compliant if it fails to satisfy one or more of the **must** level requirements for the language it implements. An implementation that satisfies all the **must** level and all the **should** level requirements for its language is said to be "unconditionally compliant"; one that satisfies all the **must** level requirements but not all the **should** level requirements for its language is said to be "conditionally compliant."

1.2 Documentation Convention

Throughout this document, the following namespace prefixes and corresponding namespace identifiers are used:

- **sul** - the SUL component namespace (http://www.openxup.org/2004/02/sul)
- **sev** - the SUL event namespace (http://www.openxup.org/2004/04/sev)
- **sres** - the SUL resource namespace (http://www.openxup.org/2004/07/sres)
- **xup** - the XUP [3] namespace (http://www.openxup.org/2004/02/xup)
- **xsd** - the XML schema [4] namespace (http://www.w3.org/2001/XMLSchema)

This is only a convention. Any namespace prefix may be used in practice.

2. Common UI Attributes and Events

This section describes UI attributes and events that are common to most UI components in SUL.

2.1 Common UI Attributes

The attributes described in this section apply to most UI components in SUL. There are UI components that only support a subset of the attributes described here; those exceptions are noted in individual component definitions.

```
%common-ui-attrs
      id = xsd:ID
      focus = xsd:boolean : false
      focus-index = xsd:nonNegativeInteger
      enabled = xsd:boolean
      visible = xsd:boolean
      fg-color = xsd:string
      bg-color = xsd:string
      bg-image = xsd:IDREF
      bg-image-layout = %image-layout : tile
      font = xsd:string
      cursor = (default | wait | crosshair | hand | text) : default
      menu = xsd:IDREF
      tooltip = xsd:string
      x = xsd:int
      y = xsd:int
      width = xsd:nonNegativeInteger
      height = xsd:nonNegativeInteger
      left-margin = xsd:nonNegativeInteger
      right-margin = xsd:nonNegativeInteger
      top-margin = xsd:nonNegativeInteger
      bottom-margin = xsd:nonNegativeInteger
      anchor = %anchor-styles : "left, top"
      dock = %layout-alignment : none
      flow-break = xsd:boolean : false
      cell = xsd:string
      column-span = xsd:positiveInteger : 1
      row-span = xsd:positiveInteger : 1
```

- **id** - the XML ID for the component.

- **focus** (optional) - whether the component is requesting for focus.
- **focus-index** (optional) - the focus index of the component. A component with lower focus index value will receive input focus before all other components with higher focus index values under the same parent.
- **enabled** (optional) - whether the component is enabled. If unspecified, the component will inherit this attribute value from its parent. If none of the ancestors specify this value, the value defaults to true.
- **visible** (optional) - whether the component is visible. If unspecified, the component will inherit this attribute value from its parent. If none of the ancestors specify this value, the value defaults to true.
- **fg-color** (optional) - the foreground color of the component. The value is either one of the sixteen predefined color values in Appendix 7, or a hexadecimal color value in the form of "#RRGGBB", where RR is the red channel, GG is the green channel, and BB is the blue channel.
- **bg-color** (optional) - the background color of the component. The possible values are the same as *fg-color*.
- **bg-image** (optional) - the background image of the component. The value refers to the ID of an image resource.
- **bg-image-layout** (optional) - the layout of the background image. The value is one of the image layout values in Appendix 13:
    - none: The image is placed in the upper-left corner of the component.
    - tile: The image is tiled across the component.
    - center: The image is centered within the component.
    - fit: The image is stretched or shrunk so that either the top and bottom sides or the left and right sides of the image will touch the component's corresponding edges. The aspect ratio of the image is maintained, and no clipping will occur.
    - fill: The image is stretched or shrunk to completely fill the component. The aspect ratio of the image is not maintained.
- **font** (optional) - the font of the component. The value is of the form "name:<font names>; size:<font size>; style:<font styles>". <font names> contains a comma-separated list of font names. Each name is either one of the three predefined generic names ("serif", "san-serif", "monospace"), or a specific font name such as "Arial". The list of font names offers a fail-over mechanism; that is, implementations should try to render the font in the order specified in <font names>, until one is found to be supported by the display system. <font size> is the size of the font, a decimal number; the default value is implementation dependant. <font styles> contains a comma-separated list of font styles. Each style value must of one of: "bold", "italic", "underline", and "strikeout". If unspecified, it implies no font style should be applied to the font.
- **cursor** (optional) - the mouse cursor when the mouse is inside the region of the component.

- **menu** (optional) - the ID of the context menu component associated with the current component.
- **tooltip** (optional) - a string containing hints or other helpful information for this component. When the user places the mouse pointer on the component, the string will be briefly displayed in a small popup window.
- **x** (optional) - the x-coordinate of the component relative to its parent container.
- **y** (optional) - the y-coordinate of the component relative to its parent container.
- **width** (optional) - the width of the component.
- **height** (optional) - the height of the component.
- **left-margin** (optional) - the left margin of the component. Margins are padding spaces outside of the component.
- **right-margin** (optional) - the right margin of the component. Margins are padding spaces outside of the component.
- **top-margin** (optional) - the top margin of the component. Margins are padding spaces outside of the component.
- **bottom-margin** (optional) - the bottom margin of the component. Margins are padding spaces outside of the component.
- **anchor** (optional) - the edges of the container to which the component is bound and how the component is resized with its parent. Anchoring the component to its parent ensures that the anchored edges remain in the same position relative to the edges of the parent when the parent is resized. The component can be anchored to one or more edges of its parent. The value of this attribute is a comma-separated list of anchor style values defined in Appendix 9:
    - none: the component is not anchored to any edges of its container.
    - left: the component is anchored to the left edge of its container.
    - right: the component is anchored to the right edge of its container.
    - top: the component is anchored to the top edge of its container.
    - bottom: the component is anchored to the bottom edge of its container.

  When a component is anchored to an edge of its container, the distance between the component and the specified edge remains constant when the container resizes. For example, if a component is anchored to the right edge of its container, the distance between the right edge of the component and the right edge of the container remains constant when the container resizes. A component can be anchored to any combination of component edges. If the component is anchored to opposite edges of its container (for example, to the top and bottom), it resizes when the container resizes. If a component has its *anchor* property set to "none", the component moves half of the distance that the container of the component is resized.

  This attribute is mutually exclusive with the *dock* attribute.

- **dock** (optional) - which side of the component is docked to its parent and how the component is automatically resized as its parent is resized. The value is one of the following layout alignment values defined in Appendix 10:
  - none: the component is not docked.
  - left: the component's left edge is docked to the left edge of its containing component.
  - right: the component's right edge is docked to the right edge of its containing component.
  - top: the component's top edge is docked to the top of its containing component.
  - bottom: the component's bottom edge is docked to the bottom of its containing component.
  - fill: all the component's edges are docked to the all edges of its containing component and sized appropriately.

  When a component is docked to an edge of its container, it is always positioned flush against that edge when the container is resized.

  This attribute is mutually exclusive with the *anchor* attribute.
- **flow-break** (optional) - whether the parent flow panel should stop laying out components in the current flow direction and wrap to the next row or column. This attribute is applicable only if the component is a child of a flow panel.
- **cell** (optional) - the cell position of the component in its parent table panel. The value is in the form of "<column>,<row>", where <column> is the column number and <row> is the row number. If the value is unspecified or "-1,-1", the component will be positioned at the first empty cell in its parent table panel. This attribute is applicable only if the component is a child of a table panel.
- **column-span** (optional) - the number of columns spanned by the component. This attribute is applicable only if the component is a child of a table panel.
- **row-span** (optional) - the number of rows spanned by the component. This attribute is applicable only if the component is a child of a table panel.

2.2 Common UI Events

The events described in this section apply to most UI components in SUL. There are UI components that only support a subset of the events described here; those exceptions are noted in individual component definitions.

All SUL events have the following event detail attributes:

```
%common-event-attrs
      descendant = xsd:IDREF
```

- **descendant** (optional) - the ID of the descendant component that fired this event. For a container component such as tree, it is more convenient to select the event only once at the tree level, rather than for each individual tree node. In this case, the value of the descendant attribute is the ID of the tree node who fired the event.

2.2.1 Action Event

Many components in SUL support action event, *sev:action*. Action events are fired when components are been acted upon. For example, a button fires the action event when it has been clicked or activated by access key.

Action events do not contain any event details.

2.2.2 Mouse Events

There are four types of mouse events: *sev:mouse-down*, *sev:mouse-up*, *sev:mouse-click*, and *sev:mouse-double-click*. Here is the event firing sequence:

1. mouse-down
2. mouse-click
3. mouse-up
4. mouse-down
5. mouse-double-click
6. mouse-up

The *mouse-click* event will fire only if user presses and then releases the mouse within the component's region. However, the *mouse-up* event will fire even if user releases the mouse outside of the component's region; in this case, the x and y values may be either negative or larger than the component's size.

2.2.2.1 Event Detail

All the mouse events have the following event details:

```
<xup:detail
        %common-event-attrs
        sev:button = xsd:string : button1
        sev:x = xsd:int
        sev:y = xsd:int
        sev:modifiers = xsd:string
/>
```

- **button** (optional) - the mouse button pressed. The value is one of the predefined values in Appendix 6.
- **x** (optional) - the x coordinate of mouse pointer.
- **y** (optional) - the y coordinate of mouse pointer.
- **modifiers** (optional) - modifier keys. The value is a white space separated list of the following modifier keys: *alt, ctrl, shift*.

2.2.2.2 Event Mask

For fine-grained event selection, all mouse evens support the following event mask:

```
<xup:mask
```

```
      sev:buttons = xsd:string : button1
      sev:modifiers = xsd:string
/>
```

- **buttons** (optional) - which mouse buttons. The value is a white space separated list of the mouse button values defined in Appendix 6. Note that the buttons are in an OR relationship. That is, the event will be fired if any of the buttons specified are used.
- **modifiers** (optional) - modifier keys. The value is a white space separated list of the following modifier keys: *alt, ctrl, shift*. Note that the keys are in an AND relationship. That is, the event will be fired only if all of the modifier keys specified are held down.

2.2.3 Keyboard Events

There are three keyboard events: *sev:key-down*, *sev:key-up*, and *sev:key-char*. Here is the event firing sequence:

1. key-down
2. key-char
3. key-up

2.2.3.1 sev:key-down and sev:key-up

2.2.3.1.1 Event Detail

The *key-down* and *key-up* events share the following event detail:

```
<xup:detail
      %common-event-attrs
      sev:key = xsd:string
      sev:modifiers = xsd:string
/>
```

- **key** - the key pressed. The value is one of the key values defined in Appendix 3.
- **modifiers** (optional) - modifier keys. The value is a white space separated list of the following modifier keys: *alt, ctrl, shift*.

2.2.3.1.2 Event Mask

The *key-down* and *key-up* events share the following event mask:

```
<xup:mask
      sev:keys = xsd:string
      sev:key-ranges = xsd:string
      sev:modifiers = xsd:string
/>
```

- **keys** (optional) - which keys will cause the event to be fired. The value is a white space separated list of the key values found in Appendix 3. Note that the keys are in an OR relationship. That is, the event will be fired if any of the keys specified are used.
- **key-ranges** (optional) - the ranges of keys that will cause the event to be fired. The value is a white space separated list of the following key range values: *letters, digits, functions, modifiers*. Note that the key ranges are in an OR relationship. That is, the

event will be fired if the key used falls into any of the specified key ranges. Key range definitions can be found in Appendix 2.

- **modifiers** (optional) - modifier keys. The value is a white space separated list of the following modifier keys: *alt, ctrl, shift*. Note that the modifier keys are in an AND relationship. That is, the event will be fired only if all of the modifier keys specified are held down.

2.2.3.2 sev:key-char

The *key-char* event is fired when a key is pressed then released. Unlike the *key-down* and *key-up* events, the *key-char* event only supports printable characters.

2.2.3.2.1 Event Detail

The *key-char* event has the following event detail:

```
<xup:detail
      %common-event-attrs
      sev:key = xsd:string
/>
```

- **key** - the key pressed. The value is one of the key values defined in Appendix 4.

2.2.3.2.2 Event Mask

The *key-char* event has the following event mask:

```
<xup:mask
      sev:keys = xsd:string
      sev:key-ranges = xsd:string
/>
```

- **keys** (optional) - which keys will cause the event to be fired. The value is a white space separated list of the key values found in Appendix 4. Note that the keys are in an OR relationship. That is, the event will be fired if any of the keys specified are used.
- **key-ranges** (optional) - the ranges of keys that will cause the event to be fired. The value is a white space separated list of the following key range values: *letters, digits*. Note that the key ranges are in an OR relationship. That is, the event will be fired if the key used falls into any of the specified key ranges. Key range definitions can be found in Appendix 2.

3. UI Components

UI components in SUL may contain other components. Components that contain other components are called **containers**. Some containers provide constraint-based layout service to its children; these containers are called **layout containers**.

3.1 Buttons

Buttons are non-container components. There are several types of buttons in SUL: <button>, <check-box>, <radio-button>, and <toggle-button>.

3.1.1 Button

The <button> element represents a push button. A button may contain a text label and / or an image.

```
%button-attrs
      %common-ui-attrs
      visual-style = %visual-styles : none
      image = xsd:IDREF
      text-align = %content-alignment : middle-center
      image-align = %content-alignment : middle-center
      access-key = (%l-letters | %digits)

<button
      %button-attrs
>
  <!-- Content: (xsd:string) -->
</button>
```

- **visual-style** (optional) - the visual style of the button. The value is one of the visual styles defined in Appendix 18. If the value is set to "theme", the appearance of the component is determined by the client's native UI environment, which may be configurable by the end user.
- **image** (optional) - the ID of an image resource.
- **text-align** (optional) - the alignment of the text label on the button. The value is one of the content alignment values in Appendix 8.
- **image-align** (optional) - the alignment of the button image. The value is one of the content alignment values in Appendix 8.
- **access-key** (optional) - the access key for activating the button.
- Content (optional) - the text label of the button. The value may contain multiple lines of text.

3.1.1.1 Events

<button> supports all common UI events. The *sev:action* event is fired when the button is clicked or activated by access key.

3.1.2 Check Box

The <check-box> element represents a check box UI component. A check box may contain a text label and / or an image. <check-box> supports all attributes of <button>.

```
<check-box
      %button-attrs
      checked = xsd:boolean : false
      check-align = %content-alignment : middle-left
>
  <!-- Content: (xsd:string) -->
</check-box>
```

- **checked** (optional) - whether the check box is in a checked state.
- **check-align** (optional) - the alignment of the check mark on the check box. The value is one of the content alignment values in Appendix 8.

- Content (optional) - the text label of the check box. The value may contain multiple lines of text.

3.1.2.1 Events
<check-box> supports all common UI events. The *sev:action* event is fired when the check box's state has changed due to end user's direct manipulation (e.g. mouse click or access key).

3.1.3 Radio Button
The <radio-button> element represents a radio button UI component. A radio button may contain a text label and / or an image. <radio-button> supports all attributes of <button>.

```
<radio-button
      %button-attrs
      checked = xsd:boolean : false
      check-align = %content-alignment : middle-left
>
  <!-- Content: (xsd:string) -->
</radio-button>
```

- **checked** (optional) - whether the radio button is in a checked state. Unlike check boxes, radio buttons are mutually exclusive. That is, for radio buttons and exclusive toggle buttons under the same parent, only one of them may be checked or selected at any time. Checking or selecting a different radio button or exclusive toggle button will cause the currently checked radio button or selected exclusive toggle button to be automatically unchecked or unselected.
- **check-align** (optional) - the alignment of the check mark on the radio button. The value is one of the content alignment values in Appendix 8.
- Content (optional) - the text label of the radio button. The value may contain multiple lines of text.

3.1.3.1 Events
<radio-button> supports all common UI events. The *sev:action* event is fired when the radio button's state has changed from false to true due to end user's direct manipulation (e.g. mouse click or access key).

3.1.4 Toggle Button
The <toggle-button> element represents a toggle button UI component. A toggle button may contain a text label and / or an image. <toggle-button> has the appearance of a regular push button but with two states. It combines the functionality of <check box> and <radio button>; that is, <toggle button> can be set to exclusive or non-exclusive. <toggle-button> supports all attributes of <button>.

```
<toggle-button
      %button-attrs
      exclusive = xsd:boolean : false
      selected = xsd:boolean : false
>
  <!-- Content: (xsd:string) -->
</toggle-button>
```

- **exclusive** (optional) - whether the toggle button is exclusive.
- **selected** (optional) - whether the toggle button is in a selected state. For radio buttons and exclusive toggle buttons under the same parent, only one of them may be checked or selected at any time. Checking or selecting a different radio button or exclusive toggle button will cause the currently checked radio button or selected exclusive toggle button to be automatically unchecked or unselected.
- Content (optional) - the text label of the toggle button. The value may contain multiple lines of text.

3.1.4.1 Events

<toggle-button> supports all common UI events. If not exclusive, the *sev:action* event is fired when the toggle button's state has changed due to end user's direct manipulation. If exclusive, the *sev:action* event is fired when the toggle button's state has changed from false to true due to end user's direct manipulation.

3.2 Label

The <label> element represents a label, which contains a read only text and / or an image.

```
<label
      %common-ui-attrs
      border = %border-styles : none
      image = xsd:IDREF
      visual-style = %visual-styles : none
      text-align = %content-alignment : top-left
      image-align = %content-alignment : middle-center
      access-key = (%l-letters | %digits)
>
  <!-- Content: (xsd:string) -->
</label>
```

- **border** (optional) - the border of the label. The value is one of the border values defined in Appendix 16.
- **visual-style** (optional) - the visual style of the label. The value is one of the visual styles defined in Appendix 18. If the value is set to "theme", the appearance of the component is determined by the client's native UI environment, which may be configurable by the end user.
- **image** (optional) - the ID of an image resource.
- **text-align** (optional) - the alignment of the text on the label. The value is one of the content alignment values in Appendix 8.
- **image-align** (optional) - the alignment of the label image. The value is one of the content alignment values in Appendix 8.
- **access-key** (optional) - the access key. When the access key is pressed, the input focus will transfer to the component next to the label, under the same parent; i.e. the component with the smallest focus index value that is greater than or equal to the label's focus index value.
- **focus** (not supported)

- **focus-index** (not supported)
- Content (optional) - the text of the label. The value may contain multiple lines of text.

3.2.1 Events

<label> supports all mouse events. It does not support any keyboard events or the *sev:action* event.

3.3 Link

The <link> element represents a hyperlink.

```
<link
      %common-ui-attrs
      border = %border-styles : none
      action = (none | view | download) : none
      href = xsd:anyURI
      text-align = %content-alignment : top-left
>
  <!-- Content: (xsd:string) -->
</link>
```

- **border** (optional) - the border of the link. The value is one of the border values defined in Appendix 16.
- **action** (optional) - the action performed when the link is activated. The value "view" means the *sev:action* event will be fired, and then the resource pointed by the *href* attribute will be opened for viewing. The value "download" means the *sev:action* event will be fired, and then the resource pointed by the *href* attribute will be downloaded. The value "none" means the *sev:action* event will be fired, and no other action will be performed.
- **href** (optional) - a URI.
- **text-align** (optional) - the alignment of the text on the link. The value is one of the content alignment values in Appendix 8.
- Content - the text of the link. The value may contain multiple lines of text.
- **menu** (not supported) - the component supports a built-in context menu which allows the end user to either view or download the resource pointed by the link.

3.3.1 Events

<link> supports all common UI events. The *sev:action* event is fired when the link is clicked or activated by keyboard.

3.4 Uploader

The <uploader> element represents a file uploader, which is capable of uploading multiple files to a designated URL on the XUP server. The upload URL is specific to server deployment and cannot be changed by the client.

The file uploader typically displays an upload button. When clicked, it shows an upload dialog which allows the end user to manage the list of files to be uploaded. In addition, the upload dialog also allows the user to start / stop the file uploading and to monitor the upload progress.

```
<uploader
      %button-attrs
      max-files = xsd:nonNegativeInteger: 0
>
  <!-- Content: (xsd:string) -->
</uploader>
```

- **max-files** (optional) - the maximum number of files that can be uploaded at one time. The value 0 implies that there is no limit.
- Content - the text label of the upload button. The value may contain multiple lines of text.

3.4.1 Events

<uploader> supports all common UI events, except the *sev:action* event.

3.5 Text

SUL supports single-line text entry (<text-field>) as well as multi-line text entry (<text-area>). Here are the common text attributes:

```
%text-attr
      %common-ui-attrs
      border = %border-styles : 3d
      max-len = xsd:int : 0
      editable= xsd:boolean : true
      text-align = %layout-alignment : left
```

- **border** (optional) - the border of the component. The value is one of the border values defined in Appendix 16.
- **max-len** (optional) - the maximum number of characters of the input text. The value "0" implies unlimited.
- **editable** (optional) - whether the text is editable.
- **text-align** (optional) - the alignment of the text. The value must be one of the following layout alignment values (defined in Appendix 10): "left", "right", and "center".

3.5.1 Text Field

The <text-field> element represents a single-line text input.

```
<text-field
      %text-attrs
      pwd-char = (%l-letters | %u-letters | %digits | %punctuations)
      auto-complete = (none | file | url | all) : none
>
  <!-- Content: (xsd:string) -->
</text-field>
```

- **pwd-char** (optional) - the character used to mask characters entered in the text field. This attribute should be specified if the component is used to input passwords.
- **auto-complete** (optional) - the auto completion behavior when the user types text. The value must be one of the following:
    - none: disables auto completion behavior.

  - file: uses local filenames as the source of auto completion.
  - url: uses URLs in browser's history as the source of auto completion.
  - all: uses both filenames and URLs as the source of auto completion.
- Content (optional) - the input text.

3.5.1.1 Events

<text-field> supports all common UI events, except the *sev:action* event. In addition, if the text field is in a dialog and the dialog has a default button, pressing the "enter" key in the text field will activate the default dialog button.

3.5.2 Text Area

The <text-area> element represents a multi-line text input.

```
<text-area
      %text-attrs
      line-wrap = xsd:boolean : true
      hscroll = xsd:boolean : false
      vscroll = xsd:boolean : false
>
  <!-- Content: (xsd:string) -->
</text-area>
```

- **line-wrap** (optional) - whether to automatically wrap words to the beginning of the next line when necessary.
- Content (optional) - the input text. The value may contain multiple lines of text.
- **hscroll** (optional) - whether the component is horizontally scrollable. However, when the *line-wrap* attribute is set to true, the text area cannot be horizontally scrolled.
- **vscroll** (optional) - whether the component is vertically scrollable.

3.5.2.1 Events

<text-area> supports all common UI events, except the *sev:action* event.

3.6 Progress Bar

The <progress-bar> element represents a progress bar, which can be used to visually indicate the progress of a lengthy operation.

```
<progress-bar
      %common-ui-attrs
      min = xsd:nonNegativeInteger: 0
      max = xsd:nonNegativeInteger : 100
      value = xsd:nonNegativeInteger : 0
      indeterminate = xsd:boolean : false
>
  <!-- Content: (##empty) -->
</progress-bar>
```

- **min** (optional) - a non-negative integer specifying the lower bound of the progress bar. The value must be less than the value of the *max* attribute.

- **max** (optional) - a non-negative integer specifying the upper bound of the progress bar. The value must be greater than the value of the *min* attribute.
- **value** (optional) - a non-negative integer specifying the current value of the progress bar. The value must be between the values of *min* and *max* attributes, inclusive.
- **indeterminate** (optional) - whether the progress bar should display a moving or rotating pattern, without indicating the real progress. The progress bar normally displays the progress of a lengthy operation based on the *value* attribute; typically, this will be rendered as continuous blocks that fill in from left to right in incremental steps. However, when this attribute is set to true, the progress bar will display a moving or rotating pattern without indicating the real progress being made (i.e. ignoring the *value* attribute).
- **focus** (not supported)
- **focus-index** (not supported)
- **font** (not supported)
- **bg-image** (not supported)
- **bg-image-layout** (not supported)

3.6.1 Events

<progress-bar> supports all mouse events, except *sev:mouse-double-click*. It does not support any keyboard events or the *sev:action* event.

3.7 Slider

The <slider> element represents is a slider component which allows the user to select a numeric value within a range by sliding the scrollable knob.

```
<slider
      %common-ui-attrs
      min = xsd:nonNegativeInteger: 0
      max = xsd:nonNegativeInteger : 10
      value = xsd:nonNegativeInteger : 0
      orient = %content-orientation : horizontal
>
  <!-- Content: (##empty) -->
</slider>
```

- **min** (optional) - a non-negative integer specifying the lower bound of the slider. The value must be less than the value of the *max* attribute.
- **max** (optional) - a non-negative integer specifying the upper bound of the slider. The value must be greater than the value of the *min* attribute.
- **value** (optional) - a non-negative integer specifying the current value of the slider. The value must be between the values of *min* and *max* attributes, inclusive.
- **orient** (optional) - whether the slider should be displayed horizontally or vertically.
- **font** (not supported)

- **fg-color** (not supported)
- **bg-image** (not supported)
- **bg-image-layout** (not supported)

3.7.1 Events

<slider> supports all mouse and keyboard events, except *sev:mouse-click* and *sev:mouse-double-click*. It does not support the *sev:action* event.

In addition, <slider> also supports the following events.

3.7.1.1 sev:state-changed

For <slider>, the *sev:state-changed* event does not support `%common-event-attrs`. This event is fired after the slider's value is changed (i.e. when the user moves the scrollable knob).

3.8 Spinner

The <spinner> element represents a component that allows the user to select a numeric value within a range. The component contains a text field and two buttons: an up button and a down button. The numeric value can be changed by clicking the up or down buttons. In addition, the user can directly enter a numeric value in the text field portion of the component.

```
<spinner
      %common-ui-attrs
      hex = xsd:boolean : false
      border = %border-styles : 3d
      min = xsd:decimal : 0
      max = xsd:decimal : 100
      value = xsd:decimal : 0
      inc = xsd:decimal : 1
      decimal-places = xsd:nonNegativeInteger: 0
      thousands-separator = xsd:boolean : false
      editable = xsd:boolean : true
      value-align = %layout-alignment : left
      control-align = %layout-alignment : right
>
  <!-- Content: (##empty) -->
</spinner>
```

- **hex** (optional) - whether the numeric value is rendered and interpreted as a hexadecimal number. This attribute controls the formatting of the following attributes: *min*, *max*, *inc*, and *value*. Note that the "0x" prefix must not be specified for hexadecimal numbers.
- **border** (optional) - the border of the component. The value is one of the border values defined in Appendix 16.
- **min** (optional) - a decimal number specifying the lower bound of the spinner. The value must be less than the value of the *max* attribute.
- **max** (optional) - an decimal number specifying the upper bound of the spinner. The value must be greater than the value of the *min* attribute.
- **value** (optional) - a decimal number specifying the current value of the spinner. The value must be between the values of *min* and *max* attributes, inclusive.

- **inc** (optional) - the amount to increment or decrement the *value* attribute when the up or down buttons in the spinner are clicked. The value must be positive.
- **decimal-places** (optional) - the number of decimal places to display in the spinner. The value must not be greater than 99.
- **thousands-separator** (optional) - whether to display thousands separators (i.e. the "," character) in the spinner when appropriate.
- **editable** (optional) - whether the numeric value can be changed by editing the text field portion of the spinner.
- **value-align** (optional) - the alignment of the numeric value in the text field portion of the spinner. The value must be one of the following layout alignment values (defined in Appendix 10): "left", "right", and "center".
- **control-align** (optional) - the alignment of the up and down buttons within the spinner. The value must be one of the following layout alignment values (defined in Appendix 10): "left" and "right".
- **bg-color** (not supported)
- **bg-image** (not supported)
- **bg-image-layout** (not supported)
- **menu** (not supported)

3.8.1 Events

<spinner> supports all mouse and keyboard events. It does not support the *sev:action* event.

In addition, <spinner> also supports the following events.

3.8.1.1 sev:state-changed

For <spinner>, the *sev:state-changed* event does not support `%common-event-attrs`. This event is fired after the spinner's value is changed (i.e. when the user clicks on the up and down buttons or direct edits the numeric value in the text field).

3.9 Date and Time

The <date-time> element displays a date and / or time value; in addition, it allows the value to be modified by the end user.

```
<date-time
      %common-ui-attrs
      format = xsd:string
      min = xsd:string
      max = xsd:string
      value = xsd:string
      calendar = xsd:boolean : false
      auto-update = xsd:nonNegativeInteger: 0
>
  <!-- Content: (##empty) -->
</date-time>
```

- **format** (optional) - the format of the date and time values. The value of this attribute should be a concatenation of date and time format specifiers defined in Appendix 19. If unspecified, the interpretation of date and time values becomes system-dependant. This may lead to unexpected behaviors; for example, a client running in a Chinese environment may have a very different default date and time format than a server running in an English environment. Note that the format specified by this attribute will be used for both interpreting and displaying date and time values.
- **min** (optional) - the minimum date and time value that can be set by the user. The format of this value must conform to the format defined by the *format* attribute. The default value for this attribute is January 1, 1753 00:00:00. The value must be less than the value of the *max* attribute, and the value cannot be less than January 1, 1753 00:00:00.
- **max** (optional) - the maximum date and time value that can be set by the user. The format of this value must conform to the format defined by the *format* attribute. The default value for this attribute is December 31, 9998 00:00:00. The value must be greater than the value of the *min* attribute, and the value cannot be greater than December 31, 9998 00:00:00.
- **value** (optional) - the date and time value. The format of this value must conform to the format defined by the *format* attribute. The value must be between the values of *min* and *max* attributes, inclusive. If unspecified, the current system time will be displayed.
- **calendar** (optional) - whether a month calendar should be displayed to allow the user to choose the date value.
- **auto-update** (optional) - how frequent (in number of seconds) the date and time value should be updated by the system clock. The date and time value will be refreshed according to the number of seconds specified by this attribute. Essentially, this allows the component to be used as a clock. The default value for this attribute is 0, which means the date and time value will not be automatically updated.
- **fg-color** (not supported)
- **bg-image** (not supported)
- **bg-image-layout** (not supported)

3.9.1 Events

<date-time> supports all mouse and keyboard events, except *sev:mouse-click* and *sev:mouse-double-click*. It does not support the *sev:action* event.

In addition, <date-time> also supports the following events.

3.9.1.1 sev:state-changed

For <date-time>, the *sev:state-changed* event does not support `%common-event-attrs`. This event is fired after the date and time value is changed by the end user.

3.10 Calendar

The <calendar> element displays a monthly calendar; in addition, it allows the user to select a range of dates within the calendar.

```
<calendar
      %common-ui-attrs
      format = xsd:string
      min = xsd:string
      max = xsd:string
      selection-start = xsd:string
      selection-end = xsd:string
      month-columns = xsd:positiveInteger: 1
      month-rows = xsd:positiveInteger: 1
>
  <!-- Content: (##empty) -->
</calendar>
```

- **format** (optional) - the format of the date values. The value of this attribute should be a concatenation of date and time format specifiers defined in Appendix 19. If unspecified, the interpretation of date values becomes system-dependant. This may lead to unexpected behaviors; for example, a client running in a Chinese environment may have a very different default date and time format than a server running in an English environment. Note that the format specified by this attribute will be used for both interpreting and displaying date values.
- **min** (optional) - the minimum date value that can be selected by the user. The format of this value must conform to the format defined by the *format* attribute. The default value for this attribute is January 1, 1753 00:00:00. The value must be less than the value of the *max* attribute, and the value cannot be less than January 1, 1753 00:00:00.
- **max** (optional) - the maximum date value that can be selected by the user. The format of this value must conform to the format defined by the *format* attribute. The default value for this attribute is December 31, 9998 00:00:00. The value must be greater than the value of the *min* attribute, and the value cannot be greater than December 31, 9998 00:00:00.
- **selection-start** (optional) - the start date of the selected range of dates. The format of this value must conform to the format defined by the *format* attribute. The value must be between the values of *min* and *max* attributes, inclusive, and it must be less than or equal to the value of *selection-end*. If unspecified, the value defaults the current system date.
- **selection-end** (optional) - the end date of the selected range of dates. The format of this value must conform to the format defined by the *format* attribute. The value must be between the values of *min* and *max* attributes, inclusive, and it must be greater than or equal to the value of *selection-start*. If unspecified, the value defaults the current system date.
- **month-columns** (optional) - the number of months to be displayed horizontally. The maximum number of months can be displayed by this component is 12 (i.e. one calendar year), so the product of this value and the value of *month-rows* must be less than or equal to 12.

- **month-rows** (optional) - the number of months to be displayed vertically. The maximum number of months can be displayed by this component is 12 (i.e. one calendar year), so the product of this value and the value of *month-columns* must be less than or equal to 12.
- **bg-color** (not supported)
- **fg-color** (not supported)
- **bg-image** (not supported)
- **bg-image-layout** (not supported)

3.10.1 Events

<calendar> supports all mouse and keyboard events, except *sev:mouse-click* and *sev:mouse-double-click*. It does not support the *sev:action* event.

In addition, <calendar> also supports the following events.

3.10.1.1 sev:selection-changed

For <calendar>, the *sev:selection-changed* event does not support `%common-event-attrs`. This event is fired after the selected range of dates is changed by the end user.

3.11 Tree

The <tree> element represents a tree, which contains a hierarchy of tree nodes.

```
<tree
      %common-ui-attrs
      border = %border-styles : 3d
      stateful = xsd:boolean : false
      auto-scroll = xsd:boolean : true
      selected = xsd:IDREF
      image = xsd:IDREF
      selected-image = xsd:IDREF
>
  <!-- Content: (##other) -->
</tree>
```

- **border** (optional) - the border of the tree. The value is one of the border values defined in Appendix 16.
- **stateful** (optional) - whether the nodes in the tree are stateful. If true, a check box should be displayed alongside each tree node.
- **auto-scroll** (optional) - whether the component will automatically display scrollbars when necessary.
- **selected** (optional) - the ID of the currently selected tree node.
- **image** (optional) - the ID of the image to be displayed alongside all tree nodes.
- **selected-image** (optional) - the ID of the image to be displayed alongside the selected tree node.
- Content (optional) - zero or more <tree-node> elements and any XML elements outside of the SUL namespace.

3.11.1 Events

<tree> supports all common UI events, except the *sev:action* event. For the *sev:key-down* and *sev:key-char* events, the value of the *descendant* attribute is the ID of the selected node before the key stroke takes effect. For the *sev:key-up* event, the value of the *descendant* attribute is the ID of the selected node after the key stroke takes effect.

In addition, <tree> also supports the following events.

3.11.1.1 sev:tree-node-expanding

For <tree>, the *sev:tree-node-expanding* event supports `%common-event-attrs`. The value of the *descendant* attribute is the ID of the tree node that is expanded.

For a tree node (in the collapsed state) with children, performing one the following actions will expand tree node, but it will NOT fire this event:

- click on the '+' icon
- double-click on the node label or image
- press the right-arrow key while the node is the currently selected node

To fire this event when the node has children, the end user must hold down the shift key while performing one of the above actions. The visual effect of node expansion will appear after firing of the event.

For a tree node without children, performing one the following actions will fire this event:

- double-click on the node label or image
- press the right-arrow key while the node is the currently selected node

In addition, this event supports the *cancelEffect* attribute of the <xup:eventResult> element. The effect of this event will be canceled when the value of *cancelEffect* is set to true; that is, the node will not be expanded.

3.11.1.2 sev:tree-node-collapsing

For <tree>, the *sev:tree-node-collapsing* event supports `%common-event-attrs`. The value of the *descendant* attribute is the ID of the tree node that is collapsed.

For a tree node (in the expanded state) with children, performing one the following actions will fire this event and then collapse the tree node:

- click on the '-' icon
- double-click on the node label or image
- press the left-arrow key while the node is the currently selected node

In addition, this event supports the *cancelEffect* attribute of the <xup:eventResult> element. The effect of this event will be canceled when the value of *cancelEffect* is set to true; that is, the node will not be collapsed.

3.11.1.3 sev:selection-changed

For <tree>, the *sev:selection-changed* event supports `%common-event-attrs`. The value of the *descendant* attribute is the ID of the newly selected tree node.

3.11.1.4 sev:state-changed

For <tree>, the *sev:state-changed* event supports `%common-event-attrs`. The value of the *descendant* attribute is the ID of the tree node that changed its state.

3.11.1.5 sev:label-edit

For <tree>, the *sev:label-edit* event is fired when the end user changes the text label of a tree node. The value of the *descendant* attribute is the ID of the tree node whose label is been changed.

```
<xup:detail
      %common-event-attrs
      sev:label = xsd:string
/>
```

- **label** - the new node label.

To edit the node label, the end user should first select the node, and then either click on the label or press the "F2" key.

This event supports the *cancelEffect* attribute of the <xup:eventResult> element. The effect of this event will be canceled when the value of *cancelEffect* is set to true; that is, the node label will not be changed.

3.12 Tree Node

The <tree-node> element represents a tree node in a tree. A tree node may in turn contain a hierarchy of descendant tree nodes.

```
<tree-node
      %common-ui-attrs
      expanded = xsd:boolean : false
      editable = xsd:boolean : false
      checked = xsd:boolean : false
      label = xsd:string
      image = xsd:IDREF
      selected-image = xsd:IDREF
>
  <!-- Content: (##other) -->
</tree-node>
```

- **expanded** (optional) - whether the tree node is in the expanded state.
- **editable** (optional) - whether the node label is editable by end users.
- **checked** (optional) - whether the tree node is checked. This attribute must not be used when the value of the *stateful* attribute of <tree> is set to false.
- **label** (optional) - the node label text.
- **image** (optional) - the ID of the image to be displayed alongside the tree node. This attribute overrides the *image* attribute of <tree>.
- **selected-image** (optional) - the ID of the image to be displayed when the tree node is selected. This attribute overrides the *selected-image* attribute of <tree>.

- Content (optional) - zero or more <tree-node> elements and any XML elements outside of the SUL namespace.
- **focus** (not supported)
- **focus-index** (not supported)
- **enabled** (not supported)
- **visible** (not supported)
- **bg-image** (not supported)
- **bg-image-layout** (not supported)
- **cursor** (not supported)
- **tooltip** (not supported)
- **x** (not supported)
- **y** (not supported)
- **width** (not supported)
- **height** (not supported)
- **left-margin** (not supported)
- **right-margin** (not supported)
- **top-margin** (not supported)
- **bottom-margin** (not supported)
- **dock** (not supported)
- **anchor** (not supported)
- **flow-break** (not supported)
- **cell** (not supported)
- **column-span** (not supported)
- **row-span** (not supported)

3.12.1 Events

<tree-node> supports all common UI events, except the *sev:action* event. For mouse events, the x and y coordinates are related to the tree component, not the node.

In addition, <tree-node> also supports the following events.

3.12.1.1 sev:tree-node-expanding

For <tree-node>, the *sev:tree-node-expanding* event does not support `%common-event-attrs`.

For a tree node (in the collapsed state) with children, performing one the following actions will expand tree node, but it will NOT fire this event:

- click on the '+' icon
- double-click on the node label or image
- press the right-arrow key while the node is the currently selected node

To fire this event when the node has children, the end user must hold down the shift key while performing one of the above actions. The visual effect of node expansion will appear after firing of the event.

For a tree node without children, performing one the following actions will fire this event:

- double-click on the node label or image
- press the right-arrow key while the node is the currently selected node

In addition, this event supports the *cancelEffect* attribute of the <xup:eventResult> element. The effect of this event will be canceled when the value of *cancelEffect* is set to true; that is, the node will not be expanded.

3.12.1.2 sev:tree-node-collapsing

For <tree-node>, the *sev:tree-node-collapsing* event does not support `%common-event-attrs`.

For a tree node (in the expanded state) with children, performing one the following actions will fire this event and then collapse the tree node:

- click on the '-' icon
- double-click on the node label or image
- press the left-arrow key while the node is the currently selected node

In addition, this event supports the *cancelEffect* attribute of the <xup:eventResult> element. The effect of this event will be canceled when the value of *cancelEffect* is set to true; that is, the node will not be collapsed.

3.12.1.3 sev:selected

For <tree-node>, the *sev:selected* event does not support `%common-event-attrs`. This event is fired after the node becomes selected.

3.12.1.4 sev:state-changed

For <tree-node>, the *sev:state-changed* event does not support `%common-event-attrs`. This event is fired after the node's state is changed (from checked to unchecked or from unchecked to checked).

3.12.1.5 sev:label-edit

For <tree-node>, the *sev:label-edit* event is fired when the end user changes the text label of the node.

```
<xup:detail
      sev:label = xsd:string
/>
```

- **label** - the new node label.

To edit the node label, the end user should first select the node, and then either click on the label or press the "F2" key.

This event supports the *cancelEffect* attribute of the <xup:eventResult> element. The effect of this event will be canceled when the value of *cancelEffect* is set to true; that is, the node label will not be changed.

3.13 Lists

SUL offers several list-based UI components. A list contains an array of items.

3.13.1 List Item

The <list-item > element represents an item within a list.

```
<list-item
      %common-ui-attrs
>
  <!-- Content: (xsd:string) -->
</list-item>
```

- Content - a string representing the text label of the item.
- **focus** (not supported)
- **focus-index** (not supported)
- **enabled** (not supported)
- **visible** (not supported)
- **fg-color** (not supported)
- **bg-color** (not supported)
- **font** (not supported)
- **bg-image** (not supported)
- **bg-image-layout** (not supported)
- **cursor** (not supported)
- **menu** (not supported)
- **tooltip** (not supported)
- **x** (not supported)
- **y** (not supported)
- **width** (not supported)
- **height** (not supported)
- **left-margin** (not supported)
- **right-margin** (not supported)
- **top-margin** (not supported)

- **bottom-margin** (not supported)
- **dock** (not supported)
- **anchor** (not supported)
- **flow-break** (not supported)
- **cell** (not supported)
- **column-span** (not supported)
- **row-span** (not supported)

3.13.1.1 Events
This component does not fire any events.

3.13.2 List Box
The <list-box> element represents a list of items displayed in a box.

```
<list-box
      %common-ui-attrs
      multi-select = xsd:boolean : false
      border = %border-styles : 3d
      selected = xsd:IDREFS
      sorted = xsd:boolean : false
>
  <!-- Content: (##other) -->
</list-box>
```

- **multi-select** (optional) - whether the component supports multiple selection.
- **border** (optional) - the border of the component. The value is one of the border values defined in Appendix 16.
- **selected** (optional) - the IDs of the selected list items.
- **sorted** (optional) - whether the list items are sorted alphabetically.
- Content - zero or more <list-item> elements and any XML elements outside of the SUL namespace.
- **bg-image** (not supported)
- **bg-image-layout** (not supported)

3.13.2.1 Events
<list-box> supports all common UI events, except the *sev:action* event. In addition, <list-box> also supports the following events.

3.13.2.1.1 sev:selection-changed
For <list-box>, the *sev:selection-changed* event is fired when the user changes the selected item(s). The event does not support `%common-event-attrs`.

3.13.3 Dropdown List
The <dropdown-list> element represents a dropdown list.

```
<dropdown-list
      %common-ui-attrs
      editable = xsd:boolean : true
      visual-style = %visual-styles : none
      visible-items = xsd:int : 8
      max-len = xsd:int : 0
      selected = xsd:IDREF
      sorted = xsd:boolean : false
      text = xsd:string
>
  <!-- Content: (##other) -->
</dropdown-list>
```

- **editable** (optional) - whether the text portion of the dropdown list is editable.
- **visual-style** (optional) - the visual style of the dropdown list. The value is one of the visual styles defined in Appendix 18. If the value is set to "theme", the appearance of the component is determined by the client's native UI environment, which may be configurable by the end user.
- **visible-items** (optional) - the maximum number of visible items in the dropdown list. The minimum value is 1 and the maximum value is 100.
- **max-len** (optional) - the maximum number of characters that can be inputted in the text portion of the dropdown list. The value "0" implies unlimited.
- **selected** (optional) - the ID of the select list item.
- **sorted** (optional) - whether the list items are sorted alphabetically.
- **text** (optional) - a string value representing the text portion of the dropdown list. This attribute cannot be used if the dropdown list is not editable.
- Content - zero or more <list-item> elements and any XML elements outside of the SUL namespace.
- **bg-image** (not supported)
- **bg-image-layout** (not supported)

3.13.3.1 Events

<dropdown-list> supports all common UI events, except the *sev:action* and *sev:mouse-double-click* events. In addition, <dropdown-list> also supports the following events.

3.13.3.1.1 sev:selection-changed

For <dropdown-list>, the *sev:selection-changed* event is fired when the user changes the selected item. The event supports `%common-event-attrs`. The value of the *descendant* attribute is the ID of the newly selected list item.

3.14 Menus

SUL offers extensive support of menus. Menus do not support many of the common UI attributes.

```
%menu-attrs
      %common-ui-attrs
```

- **focus** (not supported)
- **focus-index** (not supported)
- **fg-color** (not supported)
- **bg-color** (not supported)
- **font** (not supported)
- **bg-image** (not supported)
- **bg-image-layout** (not supported)
- **cursor** (not supported)
- **menu** (not supported)
- **tooltip** (not supported)
- **x** (not supported)
- **y** (not supported)
- **width** (not supported)
- **height** (not supported)
- **left-margin** (not supported)
- **right-margin** (not supported)
- **top-margin** (not supported)
- **bottom-margin** (not supported)
- **dock** (not supported)
- **anchor** (not supported)
- **flow-break** (not supported)
- **cell** (not supported)
- **column-span** (not supported)
- **row-span** (not supported)

SUL offers extensive support of menus. Menus do not support many of the common UI attributes.

3.14.1 Menu Bar

The <menu-bar> element represents a menu bar. It can only be attached to a window or dialog as a direct child.

```
<menu-bar
      %menu-attrs
>
  <!-- Content: (##other) -->
</menu-bar>
```

- Content - one or more <menu-item> elements and any XML elements outside of the SUL namespace.
- **enabled** (not supported)
- **visible** (not supported)

3.14.1.1 Events

<menu-bar> does not support any events.

3.14.2 Context Menu

The <context-menu> element represents a context sensitive menu. A UI component may specify a context menu via the *menu* attribute. The context menu can then be activated by right-clicking on the UI component. In addition, multiple UI components may point to the same context menu. However, a context menu must be a direct child of <window>, <dialog>, or <sul>. If the context menu is a direct child of <sul>, we call this context menu the global context menu; that is, any UI components may point to it. If the context menu is a direct child of <window> or <dialog>, only the descendants of the <window> or <dialog> and the <window> or <dialog> itself may point to it.

```
<context-menu
      %menu-attrs
>
  <!-- Content: (##other) -->
</context-menu>
```

- Content - one or more <menu-item> and / or <menu-separator> elements, and any XML elements outside of the SUL namespace.
- **enabled** (not supported)
- **visible** (not supported)

3.14.2.1 Events

<context-menu> does not support any events.

3.14.3 Menu Item

The <menu-item> element represents a menu item. In addition, a menu item may contain a submenu.

```
<menu-item
      %menu-attrs
      text = xsd:string
      stateful = xsd:boolean : false
      exclusive = xsd:boolean : false
      checked = xsd:boolean : false
      access-key = (%l-letters | %digits)
      shortcut = %keyboard-shortcuts
>
  <!-- Content: (##other) -->
</menu-item>
```

- **text** - the text displayed on the menu item.
- **stateful** (optional) - whether the menu item contains state information.

- **exclusive** (optional) - if *stateful* is true, this attribute specifies whether the menu item state is exclusive.
- **checked** (optional) - if *stateful* is true, this attribute specifies whether the menu item is in a checked state. For all sibling menu items with the *exclusive* attribute set to true, only one of them may be checked at any time. Checking a different menu item will cause the currently checked menu item to be automatically unchecked.
- **access-key** (optional) - the access key for activating the menu item.
- **shortcut** (optional) - the keyboard shortcut for activating the menu item. The value is one of the keyboard shortcuts defined in Appendix 5.
- Content - one or more <menu-item> and / or <menu-separator> elements, and any XML elements outside of the SUL namespace.

3.14.3.1 Events

<menu-item> only supports the *sev:action* event among common UI events. In addition, <menu-item> also support the *sev:context-menu-action* event.

3.14.3.1.1 sev:action

When the menu item is a descendant of a menu bar, the *sev:action* event is fired when the menu item is activated.

3.14.3.1.2 sev:context-menu-action

When the menu item is a descendant of a context menu, the *sev:context-menu-action* event is fired when the menu item is activated.

```
<xup:detail
      sev:source = xsd:IDREF
/>
```

- **source** - the ID of the UI component to which the containing context menu is associated.

3.14.4 Menu Separator

The <menu-separator> element represents a visual separator in a menu.

```
<menu-separator
      %menu-attrs
>
  <!-- Content: (##empty) -->
</menu-separator>
```

3.14.4.1 Events

<menu-separator> does not support any events.

3.15 Layout Containers

A layout container provides certain layout services to its children. Most of layout containers in SUL share the following attributes.

```
%common-layout-attrs
```

```
      %common-ui-attrs
      auto-scroll = xsd:boolean : false
```

- **auto-scroll** (optional) - whether the component will automatically display scrollbars when necessary.

3.15.1 Panel

The <panel> element represents a panel, which provides both absolute position-based and constraint-based layout services to its children.

```
%panel-attrs
      %common-layout-attrs
      border = %border-styles : none

<panel
      %panel-attrs
>
  <!-- Content: (##other) -->
</panel>
```

- **border** (optional) - the border of the panel. The value is one of the border values defined in Appendix 16.
- Content (optional) - zero or more elements in the SUL namespace (except windows, dialogs, prompts, menus, and <sul>) and any XML elements outside of the SUL namespace.

3.15.1.1 Events

<panel> supports all mouse events. It does not support any keyboard events or the *sev:action* event.

3.15.2 Flow Panel

The <flow-panel> element represents a flow panel, which arranges its children in a horizontal or vertical flow direction. Its contents can be wrapped from one row to the next, or from one column to the next. Alternatively, its contents can be clipped instead of wrapped.

The flow panel uses children's *flow-break* attribute to wrap the flow into the next row or column.

Docking and anchoring behaviors of child components differ from the behaviors in other layout containers. Both docking and anchoring are relative to the largest component in the flow direction. For vertical flow directions, the flow panel calculates the width of an implied column from the widest child in the column. All other children in this column with *anchor* or *dock* attributes are aligned or stretched to fit this implied column. For horizontal flow directions, the flow panel calculates the height of an implied row from the tallest child in the row, and all docked or anchored children in this row are aligned or sized to fit the implied row.

```
<flow-panel
      %common-panel-attrs
      dir = %content-direction : left-right
      wrap = xsd:boolean : false
>
  <!-- Content: (##other) -->
</flow-panel>
```

- **dir** - the flow direction of the flow panel. The value is one of the content direction values defined in Appendix 12.
- **wrap** - whether the flow panel should wrap its contents or let the contents be clipped.
- Content (optional) - zero or more elements in the SUL namespace (except windows, dialogs, prompts, menus, and <sul>) and any XML elements outside of the SUL namespace.

3.15.2.1 Events

<flow-panel> supports all mouse events. It does not support any keyboard events or the *sev:action* event.

3.15.3 Table Panel

The <table-panel> element represents a table panel, which arranges its contents in a grid. It can expand to accommodate new components when they are added, depending on the value of the *rows* and *columns* attributes. Setting either the *rows* or *columns* attribute to a value of 0 specifies that the table panel will be unbound in the corresponding direction.

The table panel uses children's *cell*, *column-span*, and *row-span* attributes to control child-specific layout behavior.

The anchoring behavior of child components in a table panel differs from the behavior in other containers. If the value of the child component's *anchor* attribute is set to "left" or "right", the component will be placed against the left or right border of the cell, at a distance that is equal to the component's corresponding margin attribute value. If both the "left" and "right" values are set, the component will be sized to the width of the cell, with the component's margin values taken into account. The behavior for top and bottom anchoring is analogous. If the child's *anchor* attribute has the value "none", it will be centered (both horizontally and vertically) in the cell.

```
<table-panel
      %common-panel-attrs
      cell-border = %cell-border-styles : none
      columns = xsd:int
      rows = xsd:int
      column-styles = xsd:string
      row-styles = xsd:string
>
  <!-- Content: (##other) -->
</table-panel>
```

- **cell-border** - the style of all the cell borders in the table panel. The value is one of the cell border values defined in Appendix 17.
- **columns** - the number of columns in the table panel. If unspecified or 0, the table panel may dynamically expand the number of columns.
- **rows** - the number of rows in the table panel. If unspecified or 0, the table panel may dynamically expand the number of rows.
- **column-styles** - the column styles of the table panel. The value is a comma-separated list of style values, one for each column. Each style value must be one of the following:

  - empty string: the corresponding column should auto size its width
  - non-negative integer: the width of the corresponding column in pixel
  - percentage (i.e. int%, where "int" is an integer): the width of the corresponding column expressed as a percentage of the table panel width

- **row-styles** - the row styles of the table panel. The value is a comma-separated list of style values, one for each row. Each style value must be one of the following:
  - empty string: the corresponding row should auto size its height
  - non-negative integer: the height of the corresponding row in pixel
  - percentage (i.e. int%, where "int" is an integer): the height of the corresponding row expressed as a percentage of the table panel height
- Content (optional) - zero or more elements in the SUL namespace (except windows, dialogs, prompts, menus, and <sul>) and any XML elements outside of the SUL namespace.

3.15.3.1 Events

<table-panel> supports all mouse events. It does not support any keyboard events or the *sev:action* event.

3.15.4 Tab Panel

The <tab-panel> element represents a tabbed panel in a <tab-container>. It provides both absolute position-based and constraint-based layout services to its children.

```
<tab-panel
      %common-panel-attrs
      title = xsd:string
>
  <!-- Content: (##other) -->
</tab-panel>
```

- **title** - the title of the tab panel.
- Content (optional) - zero or more elements in the SUL namespace (except windows, dialogs, prompts, menus, and <sul>) and any XML elements outside of the SUL namespace.
- **focus-index** (not supported)
- **enabled** (not supported)
- **visible** (not supported)
- **x** (not supported)
- **y** (not supported)
- **width** (not supported)
- **height** (not supported)
- **dock** (not supported)

- **anchor** (not supported)
- **flow-break** (not supported)
- **cell** (not supported)
- **column-span** (not supported)
- **row-span** (not supported)

3.15.4.1 Events

<tab-panel> supports all mouse events. It does not support any keyboard events or the *sev:action* event.

3.15.5 Shell

In SUL, the concept of shell includes windows, dialogs, and prompts; that is, UI components that appear directly on the desktop. The following is a list of common shell attributes.

```
%common-shell-attrs
      %common-layout-attrs
      title = xsd:string
      can-minimize = xsd:boolean : true
      can-maximize = xsd:boolean : true
      can-close = xsd:boolean : true
      resizable = xsd:boolean : true
      control-box = xsd:boolean : true
      state = (normal | minimized | maximized) : normal
```

- **title** - the title of the shell.
- **can-minimize** - whether the shell can be minimized by the end user.
- **can-maximize** - whether the shell can be maximized by the end user.
- **can-close** - whether the shell can be closed by the end user.
- **resizable** - whether the shell can be resized by the end user.
- **control-box** - whether to display a control box which allows the end user to minimize, maximize, close, and resize the shell.
- **state** - the display state of the shell.
- **left-margin** (not supported)
- **right-margin** (not supported)
- **top-margin** (not supported)
- **bottom-margin** (not supported)
- **dock** (not supported)
- **flow-break** (not supported)
- **cell** (not supported)
- **column-span** (not supported)

- **row-span** (not supported)

3.15.5.1 Window

The <window> element represents a window. <window> may serve as the root of a UI model, and may contain nested children windows and dialogs. A window may contain any levels of dialogs. However, there may be a maximum of two levels of windows; that is, if window 'A' contains window 'B', then 'B' must not contain further windows. We refer to 'A' as an external window, and 'B' as an internal window.

```
<window
      %common-shell-attrs
>
  <!-- Content: (##other) -->
</window>
```

- Content (optional) - zero or more elements in the SUL namespace (except <sul>) and any XML elements outside of the SUL namespace.
- **anchor** (not supported if the window is external)

3.15.5.1.1 Events

<window> supports all common UI events, except the *sev:action* event. In addition, <window> supports the *sev:window-closing* and *sev:window-closed* events. Here is the event firing sequence when the end user tries to close a window:

1. window-closing
2. window-closed

3.15.5.1.1.1 sev:window-closing

The *sev:window-closing* event is fired when the window is about to be closed by the end user. This event does not support `%common-event-attrs`.

In addition, this event supports the *cancelEffect* attribute of the <xup:eventResult> element. The effect of this event will be canceled when the value of *cancelEffect* is set to true; that is, the window will not be closed.

3.15.5.1.1.2 sev:window-closed

The *sev:window-closed* event is fired after the window is closed by the end user (disappeared from the screen). This event does not support `%common-event-attrs`.

3.15.5.2 Dialog

The <dialog> element represents a dialog. <dialog> may serve as the root of a UI model, and it may contain any levels of descendant dialogs.

Dialogs are modal; that is, a dialog receives all the mouse and keyboard input until it is closed. When a dialog is a child of <sul>, it will block input to all other windows. When a dialog is a child of a window or dialog, it will only block input to its parent window or dialog.

A dialog may contain a set of logical buttons. When one of the logical buttons is activated, the *sev:dialog-closing* event will be fired with the value of its *button* attribute set to one of the logical button names defined in Appendix 14.

```
%dialog-attrs
      %common-shell-attrs
      ok-button = xsd:IDREF
      cancel-button = xsd:IDREF
      yes-button = xsd:IDREF
      no-button = xsd:IDREF
      default-button = %dialog-buttons : none
      esc-button = %dialog-buttons : none

<dialog
      %dialog-attrs
>
  <!-- Content: (##other) -->
</dialog>
```

- **ok-button** - the ID of a <button>. When this <button> is activated, the *sev:dialog-closing* event will be fired with the value of its *button* attribute set to *ok*.
- **cancel-button** - the ID of a <button>. When this <button> is activated, the *sev:dialog-closing* event will be fired with the value of its *button* attribute set to *cancel*.
- **yes-button** - the ID of a <button>. When this <button> is activated, the *sev:dialog-closing* event will be fired with the value of its *button* attribute set to *yes*.
- **no-button** - the ID of a <button>. When this <button> is activated, the *sev:dialog-closing* event will be fired with the value of its *button* attribute set to *no*.
- **default-button** - the name of a logical button. The value is one of the logical dialog button names defined in Appendix 14. This value usually identifies the button that will have the default focus. In addition, if there is a text field in the dialog, pressing the "enter" key in the text field will activate the logical dialog button pointed to by this attribute.
- **esc-button** - the name of a logical button. The value is one of the logical dialog button names defined in Appendix 14. Pressing the "escape" key will activate the logical dialog button pointed to by this attribute.
- Content (optional) - zero or more elements in the SUL namespace (except <window> and <sul>) and any XML elements outside of the SUL namespace.
- **anchor** (not supported)

3.15.5.2.1 Events

<dialog> supports all common UI events, except the *sev:action* event. In addition, <dialog> supports the *sev:dialog-closing* and *sev:dialog-closed* events. Here is the event firing sequence when the end user tries to close a dialog:

1. dialog-closing
2. dialog-closed

3.15.5.2.1.1 sev:dialog-closing

The *sev:dialog-closing* event is fired when the dialog is about to be closed by the end user (e.g. via the dialog's close button, any of the logical dialog buttons, or keyboard shortcut). This event does not support `%common-event-attrs`.

```
<xup:detail
      sev:button = %dialog-buttons
/>
```

- **button** - the name of the logical dialog button that caused this event to be fired. The value is one of the predefined values in Appendix 14. If the event is not caused by any of the logical dialog buttons, the value is set to "none".

In addition, this event supports the *cancelEffect* attribute of the <xup:eventResult> element. The effect of this event will be canceled when the value of *cancelEffect* is set to true; that is, the dialog will not be closed.

3.15.5.2.1.2 sev:dialog-closed

The *sev:dialog-closed* event is fired after the dialog is closed by the end user (e.g. via the dialog's close button, any of the logical dialog buttons, or keyboard shortcut). Note that this event is fired only if *sev:dialog-closing* was not canceled. This event does not support `%common-event-attrs`.

```
<xup:detail
      sev:button = %dialog-buttons
/>
```

- **button** - the name of the logical dialog button that caused this event to be fired. The value is one of the predefined values in Appendix 14. If the event is not caused by any of the logical dialog buttons, the value is set to "none".

3.15.6 Prompt

Prompts are specialize dialogs that display messages and allow simple text input. The following is a list of common prompt attributes.

```
%common-prompt-attrs
      %dialog-attrs
```

- **width** (not supported)
- **height** (not supported)
- **auto-scroll** (not supported)
- **can-minimize** (not supported)
- **can-maximize** (not supported)
- **can-close** (not supported)
- **resizable** (not supported)
- **state** (not supported)
- **ok-button** (not supported)

- **cancel-button** (not supported)
- **yes-button** (not supported)
- **no-button** (not supported)
- **esc-button** (not supported)

3.15.6.1 Message Prompt

The <msg-prompt> element represents a message prompt, which contains a text message and a set of dialog buttons.

```
<msg-prompt
      %common-prompt-attrs
      button-set = %dialog-button-sets : ok
>
  <!-- Content: (xsd:string) -->
</msg-prompt>
```

- **button-set** (optional) - the set of dialog buttons to be displayed on the prompt. The value is one of the predefined dialog button set values in Appendix 15.
- Content (optional) - the message to be displayed on the prompt. The value may contain multiple lines of text.

3.15.6.1.1 Events

<msg-prompt> supports all common UI events, except the *sev:action* event. In addition, <msg-prompt> supports the *sev:dialog-closed* event.

3.15.6.1.1.1 sev:dialog-closed

The *sev:dialog-closed* event is fired after the prompt is closed by the end user (via any of the logical dialog buttons or keyboard shortcut). This event does not support `%common-event-attrs`.

```
<xup:detail
      sev:button = %dialog-buttons
/>
```

- **button** - the name of the logical dialog button that caused this event to be fired. The value is one of the predefined logical button names in Appendix 14.

3.15.6.2 Input Prompt

The <input-prompt> element represents an input prompt, which contains a text message, a one line text input field, and two dialog buttons (*ok* and *cancel*).

```
<input-prompt
      %common-prompt-attrs
      text = xsd:string
>
  <!-- Content: (xsd:string) -->
</input-prompt>
```

- **text** (optional) - the input text.

- Content (optional) - the message to be displayed on the prompt. The value may contain multiple lines of text.
- **default-button** (not supported)

3.15.6.2.1 Events

<input-prompt> supports all common UI events, except the *sev:action* event. In addition, <input-prompt> supports the *sev:dialog-closed* event.

3.15.6.2.1.1 sev:dialog-closed

The *sev:dialog-closed* event is fired after the prompt is closed by the end user (via any of the logical dialog buttons or keyboard shortcut). This event does not support `%common-event-attrs`.

```
<xup:detail
      sev:button = %dialog-buttons
/>
```

- **button** - the name of the logical dialog button that caused this event to be fired. The value is either *ok* or *cancel*.

3.16 Split Container

The <split-container> element represents a split container, which contains two UI components divided by a movable splitter. <split-container> utilizes docking to layout the children; therefore, the children's *dock* attribute must not be specified.

```
<split-container
      %common-ui-attrs
      border = %border-styles : none
      orient = %content-orientation : horizontal
      thickness = xsd:positiveInteger
>
  <!-- Content: (##other) -->
</panel>
```

- **border** (optional) - the border of the split container. The value is one of the border values defined in Appendix 16.
- **orient** (optional) - the orientation of the split container. If horizontal, the first child will be on the left hand side of the splitter and the second child will be on the right hand side of the splitter. If vertical, the first child will be placed above the splitter and the second child will be placed below the splitter.
- **thickness** (optional) - the thickness of the splitter, in number of pixels.
- Content - two elements. Each element may be either in the SUL namespace or outside of the SUL namespace. If under the SUL namespace, the element must not be windows, dialogs, prompts, menus, or <sul>.

3.16.1 Events

<split-container> does not support any events.

3.17 Tab Container

The <tab-container> element represents a group of tabbed panels.

```
<tab-container
      %common-ui-attrs
      align = %layout-alignment : top
      multiline = xsd:boolean : false
      selected = xsd:IDREF
>
  <!-- Content: (##other) -->
</panel>
```

- **align** (optional) - the alignment of the titles of the children tab panels. The value is one of the layout alignment values found in Appendix 10; the allowed values are: "top", "bottom", "left", and "right".
- **multiline** (optional) - whether the titles of the children tab panels can be displayed in multiple lines.
- **selected** (optional) - the ID of the selected child tab panel.
- Content (optional) - zero or more <tab-panel> elements and any number of XML elements outside of the SUL namespace.
- **fg-color** (not supported)
- **bg-color** (not supported)
- **bg-image** (not supported)
- **bg-image-layout** (not supported)

3.17.1 Events

<tab-container> supports all common UI events, except the *sev:action* event. In addition, it supports the *sev:selection-changed* event.

3.17.1.1 sev:selection-changed

For <tab-container>, the *sev:selection-changed* event supports `%common-event-attrs`. The value of the *descendant* attribute is the ID of the newly selected child tab panel.

3.18 Root container

The <sul> element represents the root container for a UI model. It supports multiple top level windows and dialogs by serving as their parent.

```
<sul
      %common-ui-attrs
>
  <!-- Content: (##other) -->
</sul>
```

- Content (optional) - one or more windows, dialogs, and / or prompts; zero or more context menus; and any XML elements outside of the SUL namespace.
- **focus** (not supported)

- **focus-index** (not supported)
- **enabled** (not supported)
- **visible** (not supported)
- **fg-color** (not supported)
- **bg-color** (not supported)
- **bg-image** (not supported)
- **bg-image-layout** (not supported)
- **font** (not supported)
- **cursor** (not supported)
- **menu** (not supported)
- **tooltip** (not supported)
- **x** (not supported)
- **y** (not supported)
- **width** (not supported)
- **height** (not supported)
- **left-margin** (not supported)
- **right-margin** (not supported)
- **top-margin** (not supported)
- **bottom-margin** (not supported)
- **dock** (not supported)
- **anchor** (not supported)
- **flow-break** (not supported)
- **cell** (not supported)
- **column-span** (not supported)
- **row-span** (not supported)

4. UI Resources

UI resources such as images and files share the following attributes.

```
%common-resource-attrs
      id = xsd:ID
      mime-type = xsd:string
      local-path = xsd:string
      uri = xsd:anyURI
```

- **id** - the XML ID for the component.

- **mime-type** (optional) - the MIME type of the resource.
- **local-path** (optional) - the path to a local file that contains the resource. The *uri* attribute must not be used if this attribute is used.
- **uri** (optional) - the resource URI. The *local-path* attribute must not be used if this attribute is used.

4.1 sres:image

The <image> element represents an image resource.

```
<image
      %common-resource-attrs
>
  <!-- Content: (xsd:base64Binary) -->
</image>
```

- Content (optional) - the Base64 encoded content of the resource.

4.2 sres:file

The <file> element represents a file resource.

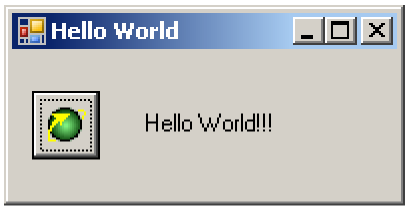
```
<file
      %common-resource-attrs
>
  <!-- Content: (xsd:base64Binary) -->
</file>
```

- Content (optional) - the Base64 encoded content of the resource.

5. Example

5.1 Hello World

In this section illustrates an example UI model, Hello World, which displays a window that contains an image button and a text label.



5.1.1 UI Resources

Here is the SUL resource description for the globe image on the button:

```
  <image xmlns="http://www.openxup.org/2004/07/sres" id="globe"
         local-path="/globe.gif"/>
```

As shown above, the image resource has an ID "globe" and the local path points to a file "/globe.gif".

5.1.2 UI Components

The following is the SUL component description:

```
<window id='wmain' title='Hello World' xmlns='http://www.openxup.org/2004/02/sul'>
  <flow-panel id='flow1' dir="horizontal" dock="fill">
    <button id='b1' image='globe'
            left-margin='10' top-margin='20' bottom-margin='20'/>
    <label id='l1' left-margin='20' right-margin='20'
           top-margin='20' bottom-margin='20'>Hello World!!!</label>
  </flow-panel>
</window>
```

Here, the main window is "wmain", and within it there is a horizontal flow panel containing a button "b1" and a label "l1". The button has an image with ID "globe", which points to the image resource defined in the last section. The label displays the text "Hello World!!!".

The flow panel "flow1" layouts out the globe button and the text label horizontally in a row. In addition, it fills up the entire window. Note that both the flow panel and the window do not specify any width or height; their size is automatically calculated.

6. Appendix

6.1 Appendix 1: Key Definitions

```
%l-letters
       a
       b
       c
       d
       e
       f
       g
       h
       i
       j
       k
       l
       m
       n
       o
       p
       q
       r
       s
       t
       u
       v
       w
       x
       y
       z

%u-letters
       A
       B
       C
       D
       E
       F
       G
       H
       I
       J
       K
       L
       M
       N
       O
       P
       Q
       R
       S
       T
       U
       V
       W
       X
       Y
       Z
```

```
%digits
      0
      1
      2
      3
      4
      5
      6
      7
      8
      9

%punctuations
      tilde
      back-quote
      excl
      at
      hash
      dollar
      percent
      caret
      amp
      asterisk
      l-parenthesis
      r-parenthesis
      l-bracket
      r-bracket
      l-brace
      r-brace
      minus
      underscore
      equal
      plus
      backslash
      pipe
      colon
      semicolon
      quot
      apos
      comma
      lt
      period
      gt
      slash
      question

%functions
      f1
      f2
      f3
      f4
      f5
      f6
      f7
      f8
      f9
      f10
      f11
      f12

%control-chars
```

```
      backspace
      enter
      esc
      space
      tab

%control-keys
      alt
      caps
      ctrl
      del
      down
      end
      home
      ins
      l-alt
      l-ctrl
      l-shift
      left
      ctx-menu
      num-lock
      pg-down
      pg-up
      pause
      r-alt
      r-ctrl
      r-shift
      right
      shift
      up
```

6.2 Appendix 2: Key Range Definitions

6.2.1 letters

```
      %l-letters
      %u-letters
```

6.2.2 digits

```
      %digits
```

6.2.3 functions

```
      %functions
```

6.2.4 modifiers

```
      alt
      ctrl
      shift
```

6.3 Appendix 3: Keys for key-down and key-up Events

```
      %l-letters
      %digits
      %functions
      %control-chars
      %control-keys
```

6.4 Appendix 4: Keys for key-char Event

```
      %l-letters
      %u-letters
      %digits
      %punctuations
      %control-chars
```

6.5 Appendix 5: Keyboard Shortcuts

```
%keyboard-shortcuts
      f1
      f2
      f3
      f4
      f5
      f6
      f7
      f8
      f9
      f10
      f11
      f12

      alt-0
      alt-1
      alt-2
      alt-3
      alt-4
      alt-5
      alt-6
      alt-7
      alt-8
      alt-9

      alt-f1
      alt-f2
      alt-f3
      alt-f4
      alt-f5
      alt-f6
      alt-f7
      alt-f8
      alt-f9
      alt-f10
      alt-f11
      alt-f12

      ctrl-0
      ctrl-1
      ctrl-2
      ctrl-3
      ctrl-4
      ctrl-5
      ctrl-6
      ctrl-7
      ctrl-8
      ctrl-9

      ctrl-a
      ctrl-b
```

```
        ctrl-c
        ctrl-d
        ctrl-e
        ctrl-f
        ctrl-g
        ctrl-h
        ctrl-i
        ctrl-j
        ctrl-k
        ctrl-l
        ctrl-m
        ctrl-n
        ctrl-o
        ctrl-p
        ctrl-q
        ctrl-r
        ctrl-s
        ctrl-t
        ctrl-u
        ctrl-v
        ctrl-w
        ctrl-x
        ctrl-y
        ctrl-z

        ctrl-f1
        ctrl-f2
        ctrl-f3
        ctrl-f4
        ctrl-f5
        ctrl-f6
        ctrl-f7
        ctrl-f8
        ctrl-f9
        ctrl-f10
        ctrl-f11
        ctrl-f12

        shift-f1
        shift-f2
        shift-f3
        shift-f4
        shift-f5
        shift-f6
        shift-f7
        shift-f8
        shift-f9
        shift-f10
        shift-f11
        shift-f12
```

6.6 Appendix 6: Mouse Button Values

```
%buttons
        button1
        button2
        button3
        button4
        button5
        button6
        button7
```

```
      button8
```

6.7 Appendix 7: Predefined Color Values

```
%colors
      black
      green
      silver
      lime
      gray
      olive
      white
      yellow
      maroon
      navy
      red
      blue
      purple
      teal
      fuchsia
      aqua
```

6.8 Appendix 8: Content Alignment

```
%content-alignment
      top-left
      top-center
      top-right
      middle-left
      middle-center
      middle-right
      bottom-left
      bottom-center
      bottom-right
```

6.9 Appendix 9: Anchor Style

```
%anchor-styles
      none
      left
      right
      top
      bottom
```

6.10 Appendix 10: Layout Alignment

```
%layout-alignment
      none
      left
      right
      top
      bottom
      center
      fill
```

6.11 Appendix 11: Content Orientation

```
%content-orientation
      horizontal
      vertical
```

6.12 Appendix 12: Content Direction

```
%content-direction
      left-right
      right-left
      top-bottom
      bottom-top
```

6.13 Appendix 13: Image Layout

```
%image-layout
      none
      tile
      center
      fit
      fill
```

6.14 Appendix 14: Dialog Buttons

```
%dialog-buttons
      none
      ok
      cancel
      yes
      no
```

6.15 Appendix 15: Dialog Button Sets

```
%dialog-button-sets
      ok
      ok-cancel
      yes-no
      yes-no-cancel
```

6.16 Appendix 16: Border Styles

```
%border-styles
      none
      single
      3d
```

6.17 Appendix 17: Cell Border Styles

```
%cell-border-styles
      none
      line
      sunken
      raised
```

6.18 Appendix 18: Visual Style Values

```
%visual-styles
      none
      flat
      popup
      theme
```

6.19 Appendix 19: Date and Time Format Specifiers

- **d**: The one- or two-digit day.

- **dd**: The two-digit day. Single-digit day values are preceded by a 0.
- **ddd**: The three-character day-of-week abbreviation.
- **dddd**: The full day-of-week name.
- **h**: The one- or two-digit hour in 12-hour format.
- **hh**: The two-digit hour in 12-hour format. Single digit values are preceded by a 0.
- **H**: The one- or two-digit hour in 24-hour format.
- **HH**: The two-digit hour in 24-hour format. Single digit values are preceded by a 0.
- **m**: The one- or two-digit minute.
- **mm**: The two-digit minute. Single digit values are preceded by a 0.
- **M**: The one- or two-digit month number.
- **MM**: The two-digit month number. Single digit values are preceded by a 0.
- **MMM**: The three-character month abbreviation.
- **MMMM**: The full month name.
- **s**: The one- or two-digit seconds.
- **ss**: The two-digit seconds. Single digit values are preceded by a 0.
- **t**: The one-letter A.M./P.M. abbreviation (A.M. is displayed as "A").
- **tt**: The two-letter A.M./P.M. abbreviation (A.M. is displayed as "AM").
- **y**: The one-digit year (2001 is displayed as "1").
- **yy**: The last two digits of the year (2001 is displayed as "01").
- **yyyy**: The full year (2001 is displayed as "2001").
- **z**: Displays the time zone offset for the system's current time zone in whole hours only. The offset is always displayed with a leading sign (zero is displayed as "+0"), indicating hours ahead of Greenwich mean time (+) or hours behind Greenwich mean time (-). The range of values is –12 to +13. If the offset is a single digit (0-9), it is displayed as a single digit with the appropriate leading sign. The setting for the time zone is specified as +X or –X where X is the offset in hours from GMT. The displayed offset is affected by daylight savings time.
- **zz**: Displays the time zone offset for the system's current time zone in whole hours only. The offset is always displayed with a leading or trailing sign (zero is displayed as "+00"), indicating hours ahead of Greenwich mean time (+) or hours behind Greenwich mean time (-). The range of values is –12 to +13. If the offset is a single digit (0-9), it is formatted with a preceding 0 (01-09) with the appropriate leading sign. The setting for the time zone is specified as +X or –X where X is the offset in hours from GMT. The displayed offset is affected by daylight savings time.

- **zzz**: Displays the time zone offset for the system's current time zone in hours and minutes. The offset is always displayed with a leading or trailing sign (zero is displayed as "+00:00"), indicating hours ahead of Greenwich mean time (+) or hours behind Greenwich mean time (-). The range of values is –12:00 to +13:00. If the offset is a single digit (0-9), it is formatted with a preceding 0 (01-09) with the appropriate leading sign. The setting for the time zone is specified as +X or –X where X is the offset in hours from GMT. The displayed offset is affected by daylight savings time.
- \c: Where *c* is any character, the escape character displays the next character as a literal.
- **'** (single quote): Quoted string. Displays the literal value of any string between two " ' " characters.
- Any other character: Other characters are interpreted as string literals.

7. References

[1] "Java AWT: Delegation Event Model", Sun Microsystems Inc., January 2004.

[2] "Key words for use in RFCs to Indicate Requirement Levels", S. Bradner, March 1997.

[3] "XUP - Extensible User Interface Protocol", <http://www.openxup.org/TR/xup/> <http://www.w3.org/TR/xup/>

[4] "XML Schema Part 1: Structures", H. Thompson, D. Beech, M. Maloney, N. Mendelsohn, October 2004.